المدرسة الوطنية للعلوم التطبيقية
École Nationale des Sciences Appliquées d'Oujda

Rapport Du Mini Projet d'application de Machine Learning
" Reconnaissance du style musical".

Réalisé Par :

Ahmed AZIZ

Mohamed BENZIZA

Imane BOULOUANE

Hafida BOUMESHOUL

Wijdane CHARKAOUI

Table de matières

1 - Introduction général

2 - Présentation du projet

3 - Source des données

4 - Analyse exploratoire des données

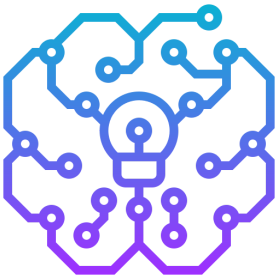
5 - Preprocessing

6 - Machine learning

1- Type d'apprentissage automatique

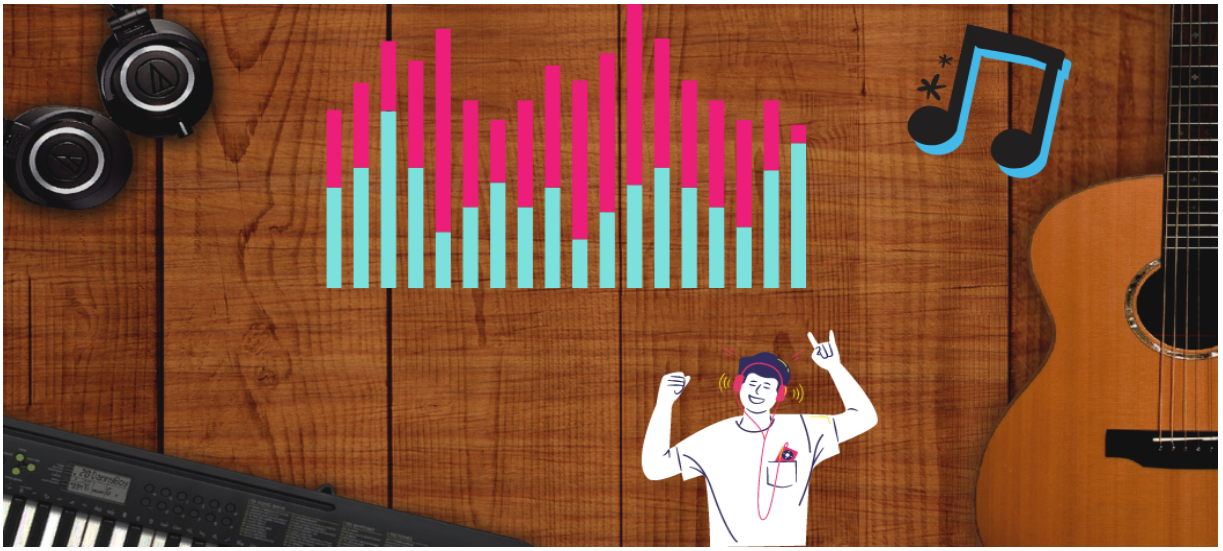
2- Algorithmes utilisés pour la résolution de la problématique

1 - Introduction général



L'apprentissage automatique, également appelé apprentissage machine, apprentissage artificiel, ou encore machine learning, est une forme d'intelligence artificielle (IA) qui permet à un système d'apprendre à partir des données et non à l'aide d'une programmation explicite. Cependant, l'apprentissage automatique n'est pas un processus simple. Au fur et à mesure que les algorithmes ingèrent les données de formation, il devient possible de créer des modèles plus précis basés sur ces données. Un modèle de machine learning est le résultat généré lorsque vous entraînez votre algorithme d'apprentissage automatique avec des données.
Le Machine Learning est de plus en plus utilisé dans l'industrie musicale, il résout des problèmes de longue date. Elle permet par exemple de supprimer facilement les bruits de microphone, de détection des genres musicaux, des enregistrements, chose qu'il était très difficile de réussir auparavant.

2 - Présentation du Projet :



Par rapport au traitement d'images et à d'autres techniques de classification, le traitement audio est l'une des tâches les plus complexes de la science des données.

"Guess It" est un mini projet d'application du machine learning qui vise à classer les fichiers audio dans certaines catégories de sons auxquelles ils appartiennent.
L'application est importante et nécessite une automatisation pour réduire l'erreur manuelle et le temps si nous devons classer la musique manuellement. Donc, pour automatiser le processus, nous utilisons des algorithmes d'apprentissage automatique .
Nous présentons des modèles multi-classification pour catégoriser l'ensemble de données musicales fourni en genres musicaux distincts. Nous avons soigneusement expérimenté 5 à 6 modèles d'apprentissage automatique pour obtenir les performances les plus élevées possibles, y compris le réglage des hyperparamètres.

3 - Source des données

Le site : www.kaggle.com nous a fournit un dataset riche et très intéressant.

UCI Machine Learning Repository : https://archive.ics.uci.edu/ml/index.php.

Le site www.medium.com nous a aidé à bien organiser le travail et les modèles.

4 - Analyse exploratoire des données

Objectif :

- Comprendre du mieux possible nos données (un petit pas en avant vaut mieux qu'un grand pas en arrière)
- Développer une première stratégie de modélisation

Checklist de base:

Analyse de Forme :

- **variable target** : music_genre
- **lignes et colonnes** : 50005, 18
- **types de variables** : qualitatives : 2, quantitatives : 11
- **Analyse des valeurs manquantes** :
- que 5 lignes NaN
- certaines valeurs sont déclarées par “?” qui n’est pas valide

Analyse de Fond :

- **Visualisation de la target** :

Il y a 10 classes (Hip-Hop, Jazz, electronic, Classical, Rap, Blues, Anime, Alternative, Country, Rock) qui contiennent 5000 labels chacune.

- **Signification des variables** :

Popularity : taux de popularité.
Acousticness : degré d'acoustique. 1.0 = chanson acoustique.
Danceability: adaptabilité à la danse. 0 = la moins dansante; 1 = la plus dansante.
Energy décrit l'intensité.
Instrumentalness : quantité de voix. 1.0 = chanson instrumentale.
liveness : probabilité que la chanson ait été enregistrée en live. Supérieure à 0,8 = morceau en live,
Speechiness : présence de mots parlés. Supérieur à 0,66 = composé de mots parlés; entre 0,33 et 0,66 = contenaient de la musique et des mots; inférieur à 0,33 = pas de paroles.
valence : décrit la positivité musicale du morceau. 0 = négatif; 1 = positif,

- **Relation Variables / Target** :
    - les features popularité, loudness, energy et acousticness sont les features qui changent le plus selon les genres.

Analyse plus détaillée:

- **Relation Variables / Variables** :

il y a une forte corrélation entre l'énergie du morceau et son intensité (corrélation positive 0,84). D'autres paramètres ont une forte corrélation négative, c'est le cas de : l'énergie et l'acoustique l'acoustique et l'intensité

hypothèses nulle (H0):

L'énergie est un facteur majeur dans la détermination du genre de la musique.

5 - Preprocessing

- Elimination des colonnes inutiles: instance_it, artist_name, track_name, obtained_date
- Elimination des ligne NAN
- Elimination des lignes qui contiennent des valeurs incompréhensibles (? dans tempo)
- Transformation des variable qualitatives en des valeurs entre 0 et n pour les normaliser

6 - Machine Learning

Maintenant que la première partie est finie, entrons dans le vif du sujet. Dans cette partie nous allons construire des modèles de classification et nous allons les comparer pour voir lequel convient le mieux.

Voici les différents algorithmes que nous avons essayé:

SVM, Decision Tree, Logistic Regression, Random Forest, KNN, Naive Bayes.

1 - SVM:

SVM est un modèle d'apprentissage automatique supervisé qui utilise des algorithmes de classification pour les problèmes de classification à deux groupes. Il a l'avantage significatif d'une vitesse élevée et de meilleures performances que les réseaux de neurones.

On a défini une fonction **model_test** qui entraîne le modèle avec fit et prédit les valeurs sur l'ensemble de test.

```
def model_test(model, title = ".."):
    model.fit(X_train, y_train)
    preds = model.predict(X_test)
    print(confusion_matrix(y_test, preds))
    print('Accuracy', title, ':', round(accuracy_score(y_test, preds), 2))
    print('Score',title,':', round(model.score(X_test,y_test),2),'\n')
```

Entraînement du modèle:

```
svm = SVC()
model_test(svm, "Support Vector Machine")
```

```
Accuracy Support Vector Machine : 0.56
Score Support Vector Machine : 0.56
```

2 - KNN: KNN C'est l'un des algorithmes d'apprentissage automatique les plus élémentaires. Il est basé sur l'idée que chaque point de données proche d'un autre appartient à la même classe.

Entraînement du modèle:

```
Accuracy KNN : 0.49
Score KNN : 0.49
```

3 - Decision Tree:

Decision Tree est une technique d'apprentissage automatique supervisée qui sépare constamment les données en fonction d'un paramètre fractionné.

Entraînement du modèle:

Decision Tree:

	precision	recall	f1-score	support
0	0.28	0.29	0.29	1119
1	0.66	0.67	0.66	1101
2	0.41	0.40	0.41	1116
3	0.74	0.75	0.75	1127
4	0.44	0.43	0.43	1132
5	0.49	0.48	0.48	1133
6	0.30	0.34	0.32	1116
7	0.41	0.39	0.40	1140
8	0.29	0.27	0.28	1156
9	0.41	0.40	0.40	1115
accuracy			0.44	11255
macro avg	0.44	0.44	0.44	11255
weighted avg	0.44	0.44	0.44	11255

4 - Logistic Regression:

La régression logistique consiste à trouver une fonction linéaire C(X) qui permette d'estimer la probabilité de Y=1 connaissant X : p(Y=1|X)=eC(X)1+eC(X). Autrement dit, cela revient à trouver une séparation linéaire des caractéristiques qui minimise un critère d'erreur.

On veut maintenant prédire le genre musical à partir de toutes les caractéristiques, et évaluer la qualité de cette prédiction en utilisant la régression logistique définie dans la librairie **sklearn** .

```
# Train our logistic regression

logreg = LogisticRegression(random_state=10)
logreg.fit(X_train,Y_train)

# Predict the labels for the test data
Y_pred_log = logreg.predict(X_test)

accuracy_score(Y_pred_log,Y_test)
```

L'accuracy est de 0.21 ce qui n'est pas satisfaisant.

Entraînement du modèle:

5 - Random Forest:

Random Forest fonctionne grâce à l'association de plusieurs arbres de décision.

Entraînement du modèle:

```
df = ensemble.RandomForestClassifier()
df.fit(X_train, Y_train)
y_rf = df.predict(X_test)
```

```
rf_score = accuracy_score(Y_test, y_rf)
print(rf_score)
```

```
0.5501554864504664
```

L'accuracy est de 0,55 ce qui n'est pas beaucoup, cela n'est pas étonnant au vu du nombre de classes que nous avons(10).

XGBoost : La méthode XGBoost est dérivée des arbres de décision, et très efficace, en particulier pour de grandes quantités de données.

	precision	recall	f1-score	support
0	0.43	0.38	0.40	895
1	0.80	0.73	0.77	882
2	0.64	0.53	0.58	907
3	0.84	0.86	0.85	899
4	0.59	0.62	0.60	923
5	0.65	0.64	0.64	903
6	0.37	0.41	0.39	878
7	0.54	0.52	0.53	930
8	0.37	0.34	0.35	910
9	0.50	0.66	0.57	877
accuracy			0.57	9004
macro avg	0.57	0.57	0.57	9004
weighted avg	0.57	0.57	0.57	9004

Classification:

```
classification_task(model, train_features, train_labels)
```

```
Accuracy: 0.797978676143936
F1 score: 0.7986318106783573
```

```
classification_task(model, val_features, val_labels)
```

```
Accuracy: 0.8918258551754775
F1 score: 0.8917539945841955
```

```
classification_task(model, test_features, test_labels)
```

```
Accuracy: 0.8818302976454909
F1 score: 0.8820557916766235
```

On a obtenu une **Accurracy** et un **f1 score** de près de 80% sur le train set
et de près de 90% sur le test set.
Dans ce cas, il semble que les échantillons de validation et de test soient assez faciles pour le modèle, et qu'il puisse facilement deviner leurs classes. De plus, ces résultats montrent que le modèle ne sous-estime ni ne surestime les échantillons d'entraînement.

6 - Naive Bayes:

Naive Bayes Les classificateurs de Bayes naïfs sont basés sur le théorème de Bayes. Le théorème de Bayes est utilisé pour créer des classificateurs de Bayes naïfs. Les hypothèses d'indépendance élevée entre les caractéristiques sont l'une des hypothèses formulées. Ces classificateurs supposent que la valeur d'une entité n'est pas liée à la valeur d'une autre caractéristique. C'est un classificateur probabiliste, qui fait des prédictions basées sur la probabilité d'un objet.

Entraînement du modèle:

```
nb = GaussianNB()
model_test(nb, "Naive Bayes")
```

```
Accuracy Naive Bayes : 0.45
Score Naive Bayes : 0.45
```

6 - AdaBoost:

AdaBoost est un méta-algorithme utilisé en association avec de nombreux autres types d'algorithmes d'apprentissage afin d'en améliorer les performances.

Entraînement du modèle:

```
Accuracy AdaBoost : 0.48
Score AdaBoost : 0.48
```

Conclusion et analyse critique des résultats:

Les premiers résultats obtenus sont plutôt intéressants même si on pouvait espérer obtenir de meilleures performances.

- **SVM :** 0.56
- **KNN:** 0.49
- **Logistic Regression :** 0.21
- **Decision Tree:** 0.44
- **Naive Bayes:** 0.45
- **Random Forest:** 0.88
- **Adaboost:** 0.48

RandomForest / XGBoost /SVM : ces modèles n'ont pas une précision assez importante.

La classification en utilisant Estimator fournit des résultats qui sont bons pour un problème possédant 10 classes.